UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER OF PATENTS AND TRADEMARKS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/492,242	01/27/2000	Andrew E. Kalman	P4300	9619

24739 7590 05/06/2003

CENTRAL COAST PATENT AGENCY
PO BOX 187
AROMAS, CA 95004

EXAMINER
TANG, KENNETH

ART UNIT	PAPER NUMBER
2127	

DATE MAILED: 05/06/2003

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)
	09/492,242	KALMAN, ANDREW E.
	Examiner	Art Unit
	Kenneth Tang	2127

-- The MAILING DATE of this communication appears on the cover she t with the correspondenc address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133).
- Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *13 February 2003* .

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-20* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-20* is/are rejected.

7)☒ Claim(s) *1* is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☒ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

11)☐ The proposed drawing correction filed on _____ is: a)☐ approved b)☐ disapproved by the Examiner.

If approved, corrected drawings are required in reply to this Office action.

12)☐ The oath or declaration is objected to by the Examiner.

Priority under 35 U.S.C. §§ 119 and 120

13)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____ .

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

14)☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. § 119(e) (to a provisional application).

a) ☐ The translation of the foreign language provisional application has been received.

15)☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. §§ 120 and/or 121.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449) Paper No(s) _____ .
4) ☐ Interview Summary (PTO-413) Paper No(s). _____ .
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other:

DETAILED ACTION

1. This final action is in response to paper number 4, Amendment A, which was received on 2/13/03. Applicant's arguments have been fully considered but they are not deemed to be persuasive. Claims 1-20 are presented for examination.

Specification

Claim 1 is objected to because of the following informalities:

- Grammatical error on line 12: "only task level" should be corrected to "only at task level";

Appropriate correction is required.

Claim Rejections - 35 USC § 103

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

1. Claims 1-20 are rejected under 35 U.S.C. 103(a) as being unpatentable over Song et al.(hereinafter Song) (US 6,061,711) in view of Bronte (US 6,061,709).

Referring to claim 1, Song teaches a real-time operating system (RTOS) *("real-time operating system", col. 4, lines 11-12)* for use with minimal-memory controllers *("minimal amount of processor state information", "minimal processor state information sub-block 614", "memory", col. 12, lines 2-10)* comprising:

- a kernel for managing task execution, including context switching *("real-time kernel 510", "scheduling tasks for execution", col. 7, lines 16-33, and "In a multi-tasking computing system environment, one program is halted and context switched out so that a processor may context switch in a subsequent program for execution", see Abstract)*;
- a plurality of defined tasks as code sets, individual ones of the tasks having subroutines for accomplishing tasks *("software", "scheduling tasks for execution", col. 7, lines 27-33, and "task instruction", col. 12, lines 11-17, and "more than two tasks", col. 15, lines 31-35, "performs general processing functions", "real-time operating system operations", col. 4, lines 10-11)); It is inherent that the "task instruction" is a set of computer code.*

Song fails to explicitly teach:

- having the subroutines callable in nested levels for accomplishing the tasks.
- characterized in that the kernel constrains context switching to occur only task level, rather than allowing context switches at lower sub-routine level.

However, Bronte teaches context switching with subroutines or "service calls" in nested levels for accomplishing tasks *("service calls", "call is nested", "from tasks", "kernel", col. 10, lines 1-13)*. It would have been obvious to one of ordinary skill in the art at the time the invention was

made to include the feature of having the subroutines callable in nested levels for accomplishing tasks for the reason of increasing the control of the system. By having scheduling context switching when the kernel is nested, the process guarantees that all scheduling occurs only once *(col. 10, lines 7-10)*. Furthermore, the reference of Bronte teaches context switching with a kernel occurring at the task level *("context switch", "kernel which changes the current running task to a new running task", "executing context to the new running task", "task control block is going to be the next to execute", col. 10, lines 28-39)*. It would have been obvious to one of ordinary skill in the art at the time the invention was made to include the feature of having the context switching to occur at the task level for the reason of maintaining the control of the system. Context switches need to be at the task level because a single start/resume address is stored per task and the task always returns to the scheduler.

Referring to claims 2 and 12, Song teaches:

- the RTOS operates with a single call return stack common to all of the defined tasks *("return address", "application program", "program execution block 602", "return address stack", "context switched", "current address" located in "vector program counter", col. 10, lines 65-67 and col. 11, lines 1-4). It is inherent in the reference of Song that the call for the soft return address to the stack during the context switching is common to all of the defined tasks.*

Referring to claims 3 and 13, Bronte teaches:

- the single stack is implemented as a general-purpose stack *("operand stack", "stack pointer", Figure 9, and col. 8, lines 35-42).*

While claims were rejected under 35 USC 112, 1st and 2nd paragraph, in order to advance prosecution, claims will be treated on the merits in view of the examiner's best understanding of the disclosure and the prior art.

Referring to claims 4 and 14, Song teaches:

- the single stack is implemented as a hardware call ...return stack *("return address stack", lines 65-67 and col. 11, lines 1-4, "bits of VISRC are set by harware", "bits are reset by software before execution of the co-processor 204 resumes", "enable bit set", "interrupt is signaled", col. 5, lines 31-40).*

Referring to claims 5 and 15, Bronte teaches:

- comprising a specific task control block assigned to each task, wherein a single task-resume address is saved *("Each task has an associated task control block that resides in system memory.", col. 10, lines 28-29).*

Referring to claims 6 and 16, Bronte teaches:

- additional task-specific information is saved *(within the "task control block", "context save", "context restore", "task", "saves the registers", col. 10, lines 40-48).*

Referring to claims 7 and 17, Song teaches:

- a task-resume address is obtained in a context switch by placing a label at the point where the task is to resume *("marked", "context switch program instruction", "multiple locations", "processor state information", "resuming execution", col. 2, lines 25-30, and "save location of context restoration subroutine sub-block 616, Figure 6, "task instruction", "offset field identifies an address of a context restoring subroutine location which coprocessor 204 will execute upon resumption of the context switched out program", col. 12, lines 11-34)*;
- obtaining the address of the label and storing that address as the task-resume address *("save location of context restoration subroutine sub-block 616, Figure 6, "task instruction", "offset field identifies an address of a context restoring subroutine location which coprocessor 204 will execute upon resumption of the context switched out program", col. 12, lines 11-34)*;

Referring to claims 8 and 18, Song teaches:

- multiple labels are used within a single task to accomplish multiple context switches ("marked", task such as a "context switch program instruction", "multiple locations", "pluralities of interspersed context switch markers", "allocated to storage of processor state information", col. 2, lines 22-46).

Referring to claims 9 and 19, Song teaches:

- a wait-on-event function characterized in that the function is called only at task-level, returns a value based on whether an event is available or not, and initiates a context switch or not based on the returned value ("context switch request detector", "context switch markers", "a request to context switch out the program", "detected context switch request", "processor state information", col. 2, lines 47-60). *In addition, the reference of Bronte teaches context switching with a kernel occurring at the task level ("context switch", "kernel which changes the current running task to a new running task", "executing context to the new running task", "task control block is going to be the next to execute", col. 10, lines 28-39)*

Referring to claims 10 and 20, Song inherently teaches:

- a wait-on-event function enclosed within a (while) loop at task level, and characterized in that the task calls the wait-on-event function in the loop and examines its return code, exiting the loop if the event is available and initiates a context switch if not, and in the event of a context switch, the task recalls the wait-on-event function after resumption, being still in the loop, and repeats this procedure until exiting the loop.

The reference of Song inherently shows the while loop through illustration by a schematic/process flow diagram of the (See Figure 6). The task calls the wait-on-event function at 602. It keeps looping between 602, 604, and 606 and exits when a positive response for a context switch request is made (at 606). In the event of a context switch, the task recalls the wait-on-event function after resumption at 610 and keeps looping until it exits the loop.

Referring to claim 11, it is rejected for the same reasons as stated in the rejection of claim 1.

Remarks

2. Applicant argues (page 9, 2nd paragraph and page 10, 4th paragraph) that the patentable difference with the prior art in this claim is that context switching is constrained to occur only at task level, rather than allowing context switches at lower sub-routine level. In response to applicant's arguments against the references individually, one cannot show nonobviousness by attacking references individually where the rejections are based on combinations of references. See *In re Keller*, 642 F.2d 413, 208 USPQ 871 (CCPA 1981); *In re Merck & Co.*, 800 F.2d 1091, 231 USPQ 375 (Fed. Cir. 1986). Examiner has already stated that Song fails to explicitly teach that the context switching is constrained to occur only at task level. However, the Examiner has shown that Bronte teaches that limitation and is obvious to combine with Song.

a. In response to applicant's argument (page 9, 3rd and 4th paragraph and page 10, paragraph 3) that the references fail to show certain features of applicant's invention, it is noted that the features upon which applicant relies (i.e., "the amount of memory available to a processor", "Program Return address being included in the processor state information", "program return address only is saved") are not recited in the rejected claim(s). Although the claims are interpreted in light of the specification, limitations

from the specification are not read into the claims. See *In re Van Geuns*, 988 F.2d 1181, 26 USPQ2d 1057 (Fed. Cir. 1993).

3. Applicant argues (pages 10 and 11, last paragraphs, and page 12, 3rd paragraph) that Bronte fails to teach the constrains in which context switching occurs only at task level, rather than allowing context switches at a lower sub-routine level. However, Examiner respectfully disagrees. Bronte further teaches the kernel performing context switching in that it may be done particularly in task space rather than in interrupt space *["may be done in task space rather than in interrupt space", col. 13, lines 33-40]*.

4. Applicant argues that it is inherent that Bronte's teaching that context switching must be supported at any level, not just the task level. Examiner has reviewed Applicant's reasoning careful but has been found unpersuasive. Neither Bronte nor Song provides evidence for this inherency. Applicant states that context switching must be supported at all levels because context switching can occur within or directly due to an interrupt, i.e. completely randomly. Neither Song nor Bronte teach that example of randomness. Using the same type of argument, one can say that if it was not done in random, interrupts can be controlled so that the context switching will occur only at task level.

5. Applicant attempts to point out that a kernel is not a module that is responsible for process/task management and it makes it easier if data communication is done at a higher level, rather than a lower one. In response, Examiner refutes that argument by clarifying the previous statement by saying a kernel is the central module of an operating system. It is the part of the operating system that loads first, and it remains in main memory. Because it stays in memory, it is important for the kernel to be as small as possible while still providing all the essential services required by other parts of the operating system and applications. Typically, the kernel is responsible for memory management, process and task management, and disk management, all performed at that high level.

6. Applicant argues that Song fails to explicitly teach wherein the RTOS operates with a single call-return stack common to all of the defined tasks. However, Examiner respectfully disagrees. The lines that Applicant references of Song merely shows that context switching occurs. It makes no reference to context switching occurring at lower sub-routine levels. Again, does teach the following:

Referring to claims 2 and 12, Song teaches:

- the RTOS operates with a single call return stack common to all of the defined tasks *("return address", "application program", "program execution block 602", "return address stack", "context switched", "current address" located in "vector program counter", col. 10, lines 65-67 and col. 11, lines 1-4).*

It is inherent in the reference of Song that the call for the soft return address to the stack during the context switching is common to all of the defined tasks. Applicant's arguments have been fully considered and have been found to be unpersuasive.

Conclusion

THIS ACTION IS MADE FINAL. Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Kenneth Tang whose telephone number is (703) 305-5334. The examiner can normally be reached on 9:00am-6:30pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, John Follansbee can be reached on (703)305-8498. The fax phone numbers for the organization where this application or proceeding is assigned are (703) 746-7239 for regular communications and (703) 746-7238 for After Final communications.

Any inquiry of a general nature or relating to the status of this application or proceeding should be directed to the receptionist whose telephone number is none.

kt
April 27, 2003

MAJID BANANKHAH
PRIMARY EXAMINER